CDHKN
ORSVZ

Najdłuższy dobry podciąg

Wskazówki do rozwiązania zadania

Podciąg „dobry” to taki podciąg, w którym występują tyle samo razy wszystkie litery od ‘A’ do k-tej litery alfabetu włącznie. Wystarczy zatem zliczyć, ile razy każda litera występuje we wprowadzonym ciągu. Jeśli na przykład $k = 3$, a litera ‘A’ występuje 4 razy, litera ‘B’ – 5 razy, zaś litera ‘C’ występuje 2 razy, to wtedy decyduje *najmniejsza* z tych ilości, czyli 2. Ponieważ mamy 3 litery alfabetu, więc najdłuższy dobry podciąg ma długość $3 \cdot 2 = 6$ liter. Zauważmy, że nie precyzujemy, jak on wygląda, ponieważ mamy podać tylko jego długość.

Przy zliczaniu ilości wystąpień liter warto pamiętać, że zmienne/stałe typu *char* zachowują się jak liczby (kody ASCII), jeśli użyte są w kontekście arytmetycznym. Na przykład jeżeli c jest zmienną tego typu odpowiadającą wielkiej literze, wtedy wartość wyrażenia $c -$ ’A’ to numer miejsca tej litery w alfabecie (od 0 do 25).

Kod programu w języku C++

```
#include <iostream>
using namespace std;

int main()
{
  int n, k;
  cin >> n >> k;
  string s; cin >> s;
  int L[26] = {0};
  for(size_t i = 0; i < s.size(); i++)
    L[s[i] - 'A']++;
  int len = n;
  for(int j = 0; j < k; j++)
    len = min(len, L[j]);
  cout << len * k << endl;
  return 0;
}
```

Inicjalizacja tablicy liczników $L[\,]$ odbywa się jeszcze na etapie kompilacji programu – w tym przypadku możliwe jest tylko użycie wartości zero.

Typ danych `size_t` to liczba całkowita bez znaku, używany przy indeksowaniu struktur w rodzaju `string`-u.

Uwagi

Użycie typu `size_t` jest wskazane, jeśli program kompilowany jest z opcją `-Wall`, czyli wszystkie ostrzeżenia (ang. *warning*) są traktowane na równi z błędami. W ten sposób kompilowane są na przykład wzorcówki rozwiązań w C++ na Szkopule.

Zamiast tablicy liczników można użyć kontenera `vector` lub `array`. Takie struktury danych mogą być inicjalizowane dowolną wartością (domniemaną wartością jest zero). W powyższym przypadku deklaracja struktury liczników mogłaby wyglądać tak:

```
vector<int> L(26);
```

lub tak:

```
array<int, 26> L;
```